REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	October 1999	Professional Paper

4. TITLE AND SUBTITLE
Effect of Complementary Processing on Navy Command and Control Software

5. FUNDING NUMBERS
PE: OMN
AN: IC000083

6. AUTHOR(S)
M. G. Ceruti, Ph.D.(1), R. C. Trout(2), and T. Lee(2)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Space and Naval Warfare Systems Center(1) Complementary Systems Inc.(2)
San Diego, CA 92152–5001 Houston, TX 77058

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Space and Naval Warfare Systems Command
San Diego, CA 92110

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENTR
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Complementary Processing (CP), is a scheduling methodology to increase software execution speed. Experiments were performed to test CP with command and control software, specifically, to measure the effect of CP on the speed of processing Navy track-message data. A 37.5-fold increase was observed in the preliminary test. Recommendations for future tests are discussed.

Published in Proceedings of COMPSAC 99 by the IEEE Computer Society, October 27-29, 1999.

14. SUBJECT TERMS
Mission Area: Command and Control
Complementary Processing (CP)
Global Command and Control System - Maritime (GCCS-M)

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	Same as Report

Fast Abstract for the Twenty-Third Annual International IEEE Computer Software and Applications Conference, COMPSAC'99, October 27-29, Phoenix, AZ, USA

Effect of Complementary Processing on Navy Command and Control Software

Marion G. Ceruti, Ph.D., Member,
IEEE Computer Society
Space and Naval Warfare Systems Center, D4121
53560 Hull Street
San Diego, CA 92152-5001, USA
+1 619 553 4068, ceruti@spawar.navy.mil

Ray C. Trout and Tse Lee
Complementary Systems Inc.
18100 Upper Bay Road, Suite 110
Houston, TX, 77058, USA
+1 281 335 9103, RayTrout@aol.com

Abstract

Complementary Processing, (CP), is a scheduling methodology to increase software execution speed. Experiments were performed to test CP with command and control software, specifically, to measure the effect of CP on the speed of processing Navy track-message data. A 37.5-fold increase was observed in this preliminary test. Recommendations for future tests are discussed.

1. Introduction

The Navy is investigating Complementary Processing, (CP), a scheduling methodology for increasing software execution speed [1], to evaluate its utility in improving the performance of command and control systems. The Global Command and Control System - Maritime (GCCS-M), includes software modules for tracking the ship positions. An experiment measured the speed of execution of the track-processing software both with and without CP.

2. Experimental methods

The experiment simulated the transmission of Link16 track messages from the Advanced Combat Direction System (ACDS) into the GCCS-M software using a network with TCP/IP. The Link16 transmitter was simulated by an IBM-compatible personal computer (PC) that was used to construct a data stream consisting of messages with information derived from actual GCCS-M track data. The PC simulator produced a stream of track data in the same format as an operational Link 16 data-format message from ACDS. The PC fed this Link16 data stream into the JMCIS Hewlett Packard (HP) Model C110 computer running the HP-UX OS version 10.2.

These data were received in the GCCS-M HP machine and processed in the same manner as in an operational scenario. The GCCS-M software parsed and processed the incoming track data and stored the processed tracks in shared memory. Tracks from the PC's Link16 simulator received time stamps - one upon receipt at the HP input port and another upon storage in the HP's shared memory to enable the calculation of the processing time for each track transaction. This procedure was repeated using the GCCS-M software in its original state, until the processing time durations were recorded for of all baseline tracks.

The choice of shared memory for data storage, as opposed to a flat-file database, was a design decision based on the performance of GCCS-M independent of CP and is not associated with CP's requirement for shared memory. The use of shared memory to store the processed track data obviated the necessity to perform multiple disk I/Os during data collection and provided a more efficient test design.

When the baseline test was complete, CP software was installed on the HP computer, but not on the PC. The GCCS-M software remained the same as it was in the baseline test. The entire test was repeated using the same Link16 data stream that was used in the baseline test. The measured processing durations with CP were compared to those without CP. The mean, mode, median and standard deviation for the time differences, both with and without CP, were calculated. A Performance Improvement Factor (PIF) was calculated to summarize the effect of CP's contribution to the processing speed.

3. Data analysis and results

The CP software affixed the time stamp upon data input and to each output record that GCCS-M wrote to shared memory at the conclusion of the GCCS-M software processing. A module that consisted of instructions to read the system clock and a write to it memory was necessary in lieu of modifications to the GCCS-M software that would have added an internal time-stamp generator. This additional software was not expected to contribute a long delay to the total processing, and therefore its execution time was ignored during these preliminary tests.

This method of time measurement allows the GCCS-M software to be tested in an unmodified and undisturbed state. B(J) is the baseline time trial with CP absent and C(J) is the test time trial with CP present.

Data were collected to determine two time intervals for each track, dt(B(J)) and dt(C(J). The baseline time interval for a track processed without CP is given by:

Data were collected to determine two time intervals for each track, dt(B(J)) and dt(C(J). The baseline time interval for a track processed without CP is given by:

$$dt(B(J)) = \text{OUTPUT } t(B(J)) - \text{INPUT } t(B(J)) \quad (1)$$

Similarly, the test time interval for a track with CP is:

$$dt(C(J)) = \text{OUTPUT } t(C(J)) - \text{INPUT } t(C(J)) \quad (2)$$

where dt signifies the time difference and t(B(J)) is the notation for the time stamp of the data, e.g. INPUT t(B(J)) is the time the track was recorded in the test data generator and OUTPUT t(B(J)) is the time that the track was recorded as it was stored in shared memory with JMCIS and without CP software.

The CP software affixes a time stamp, called INPUT t(C(J)), to each track as it is released from the data-stream generator the PC. CP also affixes a time stamp, OUTPUT t(C(J), to each track when it is stored in shared memory.

The standard deviation, S, of the sample was calculated according to the following formula:

$$S = \{(\Sigma (x_i - \langle x \rangle)^2) / (n - 1) \}^{1/2} \quad (3)$$

where x_i is the value of the i^{th} processing duration, $\langle x \rangle$ is the mean processing duration, and n is the sample size of 95 for both baseline and CP test trials.

Preliminary tests indicate a significant increase in the speed of processing 95 trials of periodic GCCS-M track data with the addition of CP. Whereas 100 trials were completed, 5 were discarded due to questionable results, such as zero processing time recorded, duplicate records, or an anomalous first time derivative that occurred for one baseline track. The statistics are summarized in Table 1.

Table 1. Summary of GCCS-M track-processing durations in μsec for data processed without and with CP.

Statistic	dt(B(J)) (Baseline trials)	dt(C(J)) (CP test trials)
Mean	661529.	17624.2
Mode	663243.	107492.
Median	661522.	5646.
S	709.265	25560.7

The PIF, the ratio of the mean processing durations without CP and with CP, is as follows:

$$PIF = \langle dt(B(J)) \rangle / \langle dt(C(J)) \rangle \quad (4)$$

where the brackets indicate the mean values from Table 1. The numerical value for the PIF was 37.5, which was obtained on the HP UX system, which does not support threads. CP is expected to increase processing speeds about 30% more than this in a thread-enabled environment.

4. Discussion and recommendations

An examination of the raw baseline data, INPUT B(J), and OUTPUT B(J), reveals dramatic evidence of the inefficiency of interrupt processing. The data for the first track were received at the HP's input port at t = 0 and were not fully processed and stored in shared memory until t = 660315 μsec. During this time interval for processing the first track, the data for all of the other tracks were received at the HP input port. Therefore, while the CPU was processing computations associated with the first (and subsequent) track(s), it was interrupted multiple times. The last track data set was received at the input port prior to the time when GCCS-M completed processing the data for the first track and prior to storing the processed data of the first track in shared memory. The last track was not fully processed until all processing of prior tracks had completed, causing the last track to have the longest processing time.

The processing durations, dt(B(J))i, exhibit a systematic increase with increasing value of i. Thus, the mean value $\langle dt(B(J)) \rangle$ is a function of the number of tracks, n. This result is expected because the processing time increases with the number of interrupts, degrading the processing efficiency roughly in a linear manner:

$$dt(B(J))i = m\,i + b \quad (5)$$

where b is a constant, i is the number of the track, and m is the slope of the linear function. In this case, m was 21 μsec, + or - 2 μsec.

Equation (5) is valid in several segments over the range for which available shared memory can accommodate additional processed data sets (e.g. more tracks). Due to the limited shared memory, more than 95 baseline trials could not be processed successfully. Equation 5 is not valid in several trials in which the GCCS-M software appeared to take much longer to process the data than in most cases. Violations of (5) occurred to a lesser extent due to the periodicity fluctuations in data inputs generated by the PC.

The distributions of processing durations for both the baseline data and the CP test trials were observed to be highly asymmetric and skewed. For example, in the CP trials, the 72 values fell below the mean whereas 23 values were above the mean. This asymmetry also is expressed in the differences between the mean, the mode and the median for each set of durations, both dt(B(J)) and dt(C(J)).

Whereas the result for baseline trials were consistent with theory, the values of the processing durations for CP trials exhibited much more variance than was expected based on the theoretical predictability of processing information frames and time periods. This is expressed in Table 1 by the large value of S equal to 25560.7 μsec., which is almost 1.5 times larger than the mean processing duration for CP. S for baseline processing durations was considerably smaller. This discrepancy between theory and experimental observation requires further investigation.

The mean processing duration of the CP and the PIF are encouraging, but these results are preliminary. Additional tests are needed to explain the following observations:

- large standard deviation for CP processing duration,
- highly unusual distribution of CP processing durations with wide variance in mean, mode, and median values,
- discontinuities in linear function of baseline track processing durations vs. track number, and
- shared-memory limitations on the number of test trials.

For further directions regarding future research, see [1].

Reference

[1] M.G. Ceruti, R.C. Trout, and T. Lee, "Complementary processing and its impact on software performance," *Proceedings (Preliminary) of the Fourth International IEEE Workshop on Object-oriented Real-time Dependable Systems, WORDS'99*, pp. 171-176, Jan. 1999.